**CHAPTER 3**

**PYTHON FLOW CONTROL**

# **Python if...else Statement**

## What are if...else statement in Python?

## Decision making is required when we want to execute a code only if a certain condition is satisfied. The if…elif…else statement is used in Python for decision making.

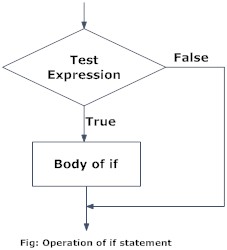
### Python if Statement Syntax

if test expression:

statement(s)

Here, the program evaluates the test expression and will execute statement(s) only if the text expression is True. If the text expression is False, the statement(s) is not executed. In Python, the body of the if statement is indicated by the indentation. Body starts with an indentation and the first unindented line marks the end. Python interprets non-zero values as True. None and 0 are interpreted as False.

### Python if Statement Flowchart



### Example: Python if Statement

### 

### When you run the program, the output will be:

3 is a positive number

This is always printed

This is also always printed.

In the above example, num > 0 is the test expression. The body of if is executed only if this evaluates to True. When variable num is equal to 3, test expression is true and body inside body of if is executed. If variable num is equal to -1, test expression is false and body inside body of if is skipped. The print() statement falls outside of the if block (unindented). Hence, it is executed regardless of the test expression.

## Python if...else Statement

### Syntax of if...else

if test expression:

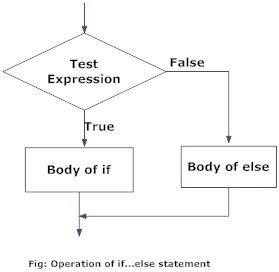
Body of if

else:

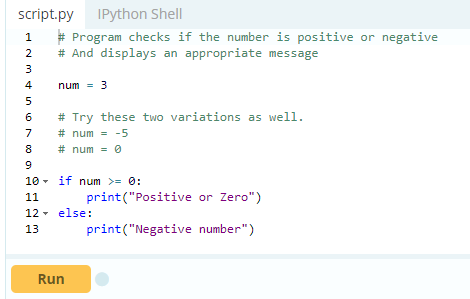
Body of else

The if..else statement evaluates test expression and will execute body of if only when test condition is True. If the condition is False, body of else is executed. Indentation is used to separate the blocks.

### Python if..else Flowchart



### Example of if...else



In the above example, when num is equal to 3, the test expression is true and body of if is executed and body of else is skipped.

If num is equal to -5, the test expression is false and body of else is executed and body of if is skipped.

If num is equal to 0, the test expression is true and body of if is executed and body of else is skipped.

## Python if...elif...else Statement

### Syntax of if...elif...else

if test expression:

Body of if

elif test expression:

Body of elif

else:

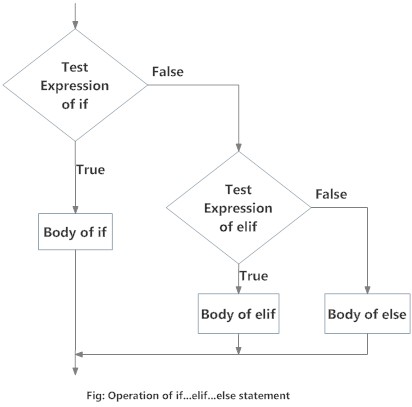
Body of else

The elif is short for else if. It allows us to check for multiple expressions. If the condition for if is False, it checks the condition of the next elif block and so on. If all the conditions are False, body of else is executed.

Only one block among the several if...elif...else blocks is executed according to the condition.

The if block can have only one else block. But it can have multiple elif blocks.

### Flowchart of if...elif...else



### Example of if...elif...else

### 

When variable num is positive, Positive number is printed.

If num is equal to 0, Zero is printed.

If num is negative, Negative number is printed

## Python Nested if statements

We can have a if...elif...else statement inside another if...elif...else statement. This is called nesting in computer programming. Any number of these statements can be nested inside one another. Indentation is the only way to figure out the level of nesting. This can get confusing, so must be avoided if we can.

### Python Nested if Example

### 

Output 1

Enter a number: 5

Positive number

Output 2

Enter a number: -1

Negative number

Output 3

Enter a number: 0

Zero

# **Python for Loop**

## What is for loop in Python?

The for loop in Python is used to iterate over a sequence ([list](https://www.programiz.com/python-programming/list), [tuple](https://www.programiz.com/python-programming/tuple), [string](https://www.programiz.com/python-programming/string)) or other iterable objects. Iterating over a sequence is called traversal.

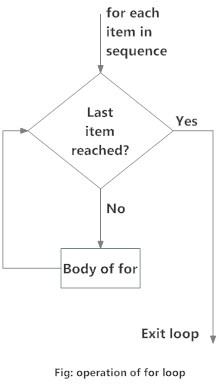
### Syntax of for Loop

for val in sequence:

Body of for

Here, val is the variable that takes the value of the item inside the sequence on each iteration. Loop continues until we reach the last item in the sequence. The body of for loop is separated from the rest of the code using indentation.

### Flowchart of for Loop



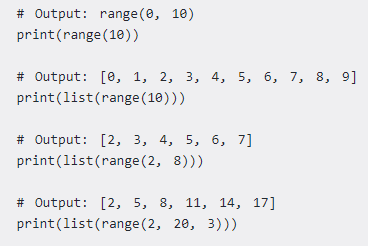
### Example: Python for Loop

### 

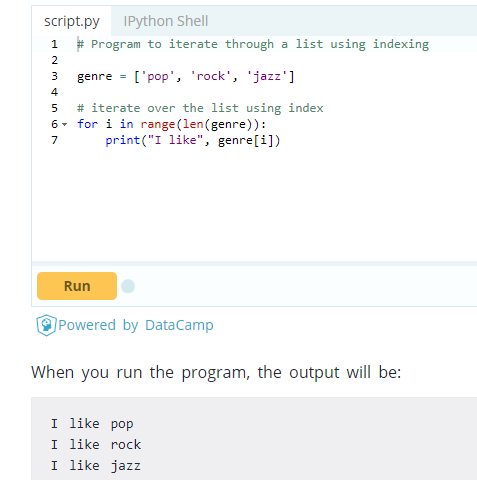
## The range() function

We can generate a sequence of numbers using range() function. range(10) will generate numbers from 0 to 9 (10 numbers). We can also define the start, stop and step size as range(start,stop,step size). step size defaults to 1 if not provided.

This function does not store all the values in memory, it would be inefficient. So it remembers the start, stop, step size and generates the next number on the go. To force this function to output all the items, we can use the function list(). The following example will clarify this.



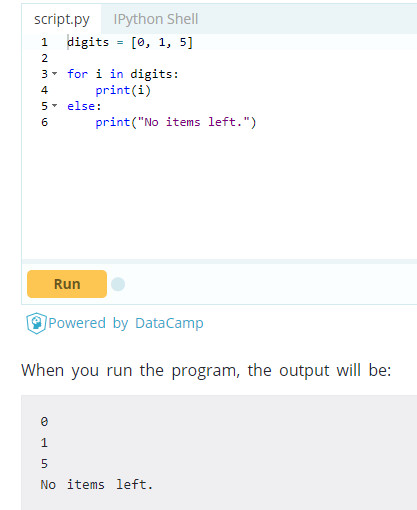
We can use the range() function in for loops to iterate through a sequence of numbers. It can be combined with the len() function to iterate though a sequence using indexing. Here is an example.



## for loop with else

A for loop can have an optional else block as well. The else part is executed if the items in the sequence used in for loop exhausts. Break statement can be used to stop a for loop. In such case, the else part is ignored. Hence, a for loop's else part runs if no break occurs.

Here is an example to illustrate this. Here, the for loop prints items of the list until the loop exhausts. When the for loop exhausts, it executes the block of code in the else and prints No items left.



# **Python while Loop**

## What is while loop in Python?

The while loop in Python is used to iterate over a block of code as long as the test expression (condition) is true. We generally use this loop when we don't know beforehand, the number of times to iterate.

### Syntax of while Loop in Python

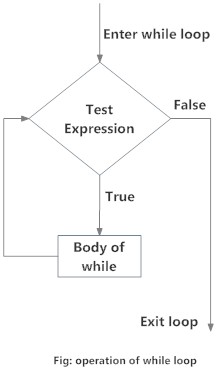
while test\_expression:

Body of while

In while loop, test expression is checked first. The body of the loop is entered only if the test\_expression evaluates to True. After one iteration, the test expression is checked again. This process continues until the test\_expression evaluates to False.

In Python, the body of the while loop is determined through indentation. Body starts with indentation and the first unintended line marks the end. Python interprets any non-zero value as True. None and 0 are interpreted as False.

### Flowchart of while Loop



### Example: Python while Loop

### 

When you run the program, the output will be:

Enter n: 10

The sum is 55

In the above program, the test expression will be True as long as our counter variable i is less than or equal to n (10 in our program). We need to increase the value of counter variable in the body of the loop. This is very important (and mostly forgotten). Failing to do so will result in an infinite loop (never ending loop). Finally the result is displayed.

## while loop with else

Same as that of [for loop](https://www.programiz.com/python-programming/for-loop), we can have an optional else block with while loop as well. The else part is executed if the condition in the while loop evaluates to False. The while loop can be terminated with a [break statement](https://www.programiz.com/python-programming/break-continue). In such case, the else part is ignored. Hence, a while loop's else part runs if no break occurs and the condition is false. Here is an example to illustrate this.



Here, we use a counter variable to print the string Inside loop three times.

On the forth iteration, the condition in while becomes False. Hence, the else part is executed.

# **Python break and continue**

## What is the use of break and continue in Python?

In Python, break and continue statements can alter the flow of a normal loop. Loops iterate over a block of code until test expression is false, but sometimes we wish to terminate the current iteration or even the whole loop without checking test expression. The break and continue statements are used in these cases

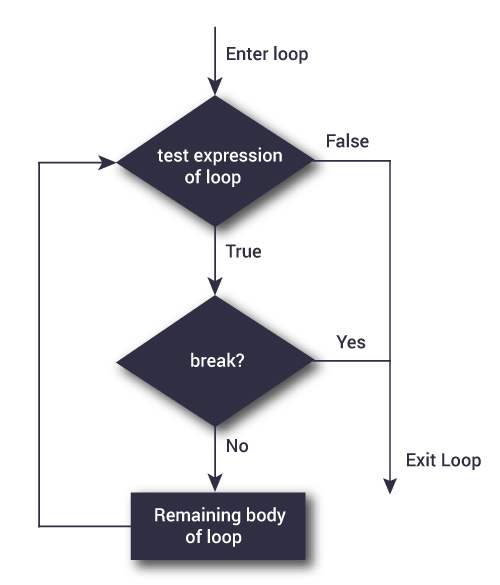
**Python break statement**

The break statement terminates the loop containing it. Control of the program flows to the statement immediately after the body of the loop. If break statement is inside a nested loop (loop inside another loop), break will terminate the innermost loop.

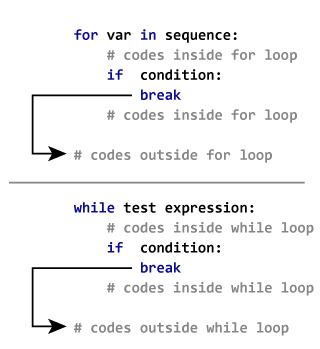
### Syntax of break

break

### Flowchart of break



The working of break statement in [for loop](https://www.programiz.com/python-programming/for-loop) and [while loop](https://www.programiz.com/python-programming/while-loop) is shown below.



### Example: Python break

### 

In this program, we iterate through the "string" sequence. We check if the letter is "i", upon which we break from the loop. Hence, we see in our output that all the letters up till "i" gets printed. After that, the loop terminates.

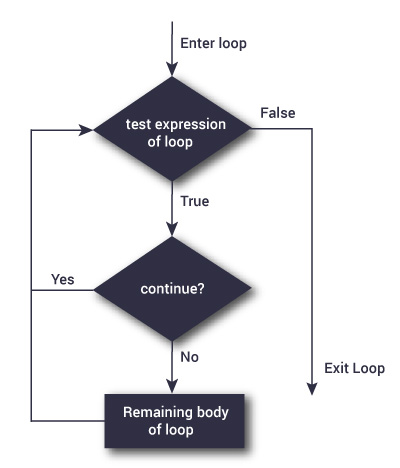
## Python continue statement

The continue statement is used to skip the rest of the code inside a loop for the current iteration only. Loop does not terminate but continues on with the next iteration.

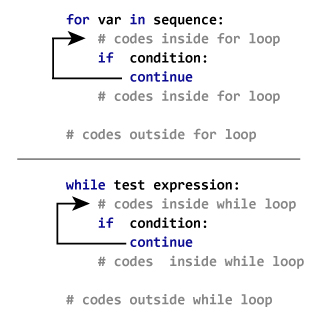
### Syntax of Continue

continue

### Flowchart of continue



The working of continue statement in for and while loop is shown below.



### Example: Python continue

### 

This program is same as the above example except the break statement has been replaced with continue. We continue with the loop, if the string is "i", not executing the rest of the block. Hence, we see in our output that all the letters except "i" gets printed

# **Python pass statement**

## What is pass statement in Python?

In Python programming, pass is a null statement. The difference between a [comment](https://www.programiz.com/python-programming/statement-indentation-comments)and pass statement in Python is that, while the interpreter ignores a comment entirely, pass is not ignored. However, nothing happens when pass is executed. It results into no operation (NOP).

### Syntax of pass

pass

We generally use it as a placeholder. Suppose we have a [loop](https://www.programiz.com/python-programming/for-loop)or a [function](https://www.programiz.com/python-programming/function) that is not implemented yet, but we want to implement it in the future. They cannot have an empty body. The interpreter would complain. So, we use the pass statement to construct a body that does nothing.

### Example: pass Statement

### 

We can do the same thing in an empty function or [class](https://www.programiz.com/python-programming/class) as well.

def function(args):

pass

class example:

pass